Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
S1	0	continu$5 adj preorder adj traversal	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/11 11:45
S2	101	(preorder pre-order (pre adj order)) adj traversal	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/11 12:33
S3	21	((child adj pointer) and (sibling adj pointer)) and counter	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/11 11:54
S4	0	"6597957".URPN.	USPAT	OR	ON	2004/02/11 11:56
S5	24	("5032979" | "5101402" | "5278901" | "5414833" | "5448724" | "5467268" | "5488715" | "5524238" | "5557742" | "5606668" | "5621889" | "5640319" | "5699513" | "5737728" | "5793763" | "5796942" | "5798706" | "5805801" | "5826014" | "5889993" | "5919257" | "5931946" | "5943652" | "5991881").PN.	USPAT	OR	ON	2004/02/11 11:56
S6	0	partial adj ((preorder pre-order (pre adj order)) adj traversal)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/11 12:04
S7	0	abbreviated adj continuation adj node	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/11 12:08
S8	120	process adj family	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/11 12:26
S9	15	(process adj family) and tree	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/11 12:26
S10	4	((preorder pre-order (pre adj order)) adj traversal) with (start stop)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/11 12:38
S11	0	abstract adj data adj type adj tree	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/11 12:38
S12	0	"6597957".URPN.	USPAT	OR	ON	2004/02/11 13:46

S13	24	("5032979" | "5101402" | "5278901" | "5414833" | "5448724" | "5467268" | "5488715" | "5524238" | "5557742" | "5606668" | "5621889" | "5640319" | "5699513" | "5737728" | "5793763" | "5796942" | "5798706" | "5805801" | "5826014" | "5889993" | "5919257" | "5931946" | "5943652" | "5991881").PN.	USPAT	OR	ON	2004/02/11 13:46
S14	62	(child adj pointer) and (sibling adj pointer)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/11 13:47
S15	58	((child adj pointer) and (sibling adj pointer)) and (@rlad<="20001031" @ad<="20001031")	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2005/02/17 13:15
S16	0	"6662358".URPN.	USPAT	OR	ON	2004/02/11 13:58
S17	27	("4520441" | "4703417" | "4841439" | "4866599" | "4868738" | "5003458" | "5047919" | "5355487" | "5408650" | "5611061" | "5613118" | "5764944" | "5768500" | "5784554" | "5802371" | "5828883" | "5928369" | "5933640" | "5940871" | "5948112" | "6002872" | "6055492" | "6077312" | "6158024" | "6282701" | "6332117" | "6349406").PN.	USPAT	OR	ON	2004/02/11 13:58
S18	12	"5682497".URPN.	USPAT	OR	ON	2004/02/11 14:25
S19	0	updated adj continuation adj node	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/11 15:01
S20	301	updated adj node	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/11 15:02
S21	111	(updated adj node) and tree	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/11 15:02
S22	79	((updated adj node) and tree) and (@rlad<="20001031" @ad<="20001031")	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2005/02/15 15:54
S23	2883	mainframe and process and tree	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/12 08:04

S24	185	mainframe and process and (tree near travers$3)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/12 08:04
S25	119	(mainframe and process and (tree near travers$3)) and (@rlad<="20001031" @ad<="20001031")	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/12 09:10
S26	32	display$3 adj process adj state	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/12 09:10
S27	11231111	1and (@rlad<="20001031" @ad<="20001031")	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/12 09:20
S28	27	(display$3 adj process adj state) and (@rlad<="20001031" @ad<="20001031")	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/12 09:12
S29	1	(preorder adj tree adj traversal) and (@rlad<="20001031" @ad<="20001031")	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/12 09:13
S30	5	(pre-order adj tree adj traversal) and (@rlad<="20001031" @ad<="20001031")	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/12 09:14
S31	1	(dynamic adj binary adj tree) and (@rlad<="20001031" @ad<="20001031")	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/12 09:20
S32	1027	718/100,101.ccls.	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/12 09:20
S33	141	718/100,101.ccls. and tree	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/12 09:20
S34	117	(718/100,101.ccls. and tree) and (@rlad<="20001031" @ad<="20001031")	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/12 12:56
S35	19	A-series adj system	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/12 11:42

S36	15	partial adj traversal	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/12 12:56
S37	3939	node with (counter timestamp)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/12 12:56
S38	3243	(node with (counter timestamp)) and (@rlad<="20001031" @ad<="20001031")	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/12 12:59
S39	128	((node with (counter timestamp)) and (@rlad<="20001031" @ad<="20001031")) and traversal	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/12 14:59
S40	31	(general adj tree) and (binary adj tree)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/12 15:00
S41	28	((general adj tree) and (binary adj tree)) and (@rlad<="20001031" @ad<="20001031")	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/02/12 15:00
S44	38	"4823310"	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2005/02/15 15:45
S45	125	((preorder or pre-order) adj traversal)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2005/02/15 15:53
S46	65	S45 and (@rlad<="20001031" @ad<="20001031")	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2005/02/15 16:32
S47	1	(dynamic adj binary adj tree) and (@rlad<="20001031" @ad<="20001031")	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2005/02/15 16:33
S48	48	(binary adj tree).ti. and (@rlad<="20001031" @ad<="20001031")	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2005/02/15 16:57
S49	299	(sibling adj node) and (@rlad<="20001031" @ad<="20001031")	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2005/02/15 16:58

S50	14	S49 and (preorder pre-order)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2005/02/17 10:44
S51	18	family with (related adj process)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2005/02/17 10:48
S52	46	binary adj tree adj represent	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2005/02/17 10:48
S53	19	S52 and (@rlad<="20001031" @ad<="20001031")	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2005/02/17 12:18
S54	1	"20030143917"	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2005/02/17 13:14
S55	0	(birary adj tree) with traversal	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2005/02/17 13:15
S56	100	(binary adj tree) with traversal	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2005/02/17 13:15
S57	43	S56 and (@rlad<="20001031" @ad<="20001031")	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2005/02/17 15:46
S58	292	(parent adj pointer) and (@rlad<="20001031" @ad<="20001031")	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2005/02/18 11:45
S59	1757	command adj queue	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2005/02/18 11:45
S60	129	instruction with (command adj queue)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2005/02/18 11:49
S61	99	S60 and (@rlad<="20010515" @ad<="20010515")	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2005/02/18 11:49

CiteSeer Find: [ ] Documents Citations

Searching for **general tree and binary tree**.
Restrict to: Header Title Order by: Expected citations Hubs Usage Date Try: Google (CiteSeer) Google (Web) Yahoo! MSN CSB DBLP
37 documents found. **Order: number of citations.**

Constructing Trees in Parallel - Atallah, Kosaraju, Larmore, Miller.. (1989) (Correct) (12 citations)
n processor parallel algorithm is given for the **general tree** construction problem. We also give an O(log values l 1 l n ,construct an ordered **binary tree** with n leaves whose levels when read form left
www.cs.unlv.edu/~larmore/Research/milkosattteng.ps.gz

Frame Representations for Texture Segmentation - Laine, Fan (1996) (Correct) (12 citations)
(DWPT) 15]16]which corresponds to a **general tree**-structured filter bank. In general, the DWPT levels. Figure 1 shows a general DWPF as a **binary tree** for a three level decomposition. For each case
ftp.cise.ufl.edu/pub/faculty/laine/texture_segm.ps

Improving Statistical Language Model Performance with.. - McMahon, Smith (1996) (Correct) (8 citations)
tags canbe considereda sub-class of the more **general tree**-based statistical language model of Bahl et of the similarity-based approach, but uses a **binary tree** representation for words and combines models
www.cs.qub.ac.uk/CS/People/home/J.McMahon/cl.ps

Nearest Common Ancestors: A survey and a new.. - Alstrup, Gavoille.. (2002) (Correct) (5 citations)
ancestor queries work using some mapping from a **general tree** to a completely balanced **binary tree**. However, is that a solution for complete balanced **binary trees** is straightforward. Furthermore, for complete
www.math.tau.ac.il/~haimk/papers/lnca-submitted.ps

Unsolved Problems Concerning Random Walks On Trees - Lyons, Pemantle, Peres (1996) (Correct) (4 citations)
Unsolved Problems concerning Random Walks on **Trees**, Classical **and** Modern Branching Processes, K. Problems concerning Random Walks on **Trees**, Classical **and** Modern Branching Processes, K. Athreya **and** P. appear Unsolved Problems Concerning Random Walks On **Trees** Russell Lyons, Robin Pemantle, Yuval Pere
www.math.ohio-state.edu/~pemantle/papers/opentree.ps

Unsolved Problems Concerning Random Walks on Trees - Lyons, Pemantle, Peres (1998) (Correct) (4 citations)
critical parameter for transience of RW on a **general tree** is exactly the exponential of the Hausdorff on multitype Galton-Watson trees. Example 2.1. **Binary tree** with pipes: Let T be a **binary tree** to every
php.indiana.edu/~rdlyons/gz/opentree.ps.gz

Gaïa: A Package for the Random Generation of Combinatorial .. - Zimmermann (1994) (Correct) (3 citations)
binary trees fC =Prod(Z Sequence(C)g plane **general trees** fD =Set(Cycle(Z)g permutations fE = generate a random object, for example a random **binary tree**: maple ?with(gaia)**binarytree** :B =
ftp.loria.fr/pub/loria/eureca/articles/Zimmermann/gaia.ps.gz

Gator: An Optimized Discrimination Network for Active Database.. - Eric Hanson (1993) (Correct) (3 citations)
TREAT/Rete) network. Gator networks are **general tree** structures. Rete and TREAT networks are a given rule, and Rete networks are limited to **binary-tree** structures. In many situations the optimal
ftp.cis.ufl.edu/pub/tech-reports/tr93/tr93-036.ps.Z

Optimized Rule Condition Testing in Ariel using Gator Networks - Eric Hanson (1995) (Correct) (2 citations)
(Generalized Treat/Rete)Gator networks are **general tree** structures. Rete and TREAT networks are a given rule, and Rete networks are limited to **binary-tree** structures. It has been observed in a
ftp.cis.ufl.edu/cis/tech-reports/tr95/tr95-027.ps

A New Tree based Domain Extension of UOWHF - Mridul Nandi Applied (2003) (Correct) (1 citation)
that the same condition becomes sucient for **general tree** based domain extension. Route of the paper : 21st June ,2003 Abstract We present a new **binary tree** based parallel algorithm for extending the
eprint.iacr.org/2003/142.ps.gz

High-Density Model for Server Allocation and Placement - Cameron, Low, Wei (2002) (Correct) (1 citation)
N) adaptive algorithm to place one server in a **general tree** and an O(log 3 N) adaptive algorithm to algorithm to place two servers in a complete **binary tree** are presented in [1]Approximation algorithms
parapet.ee.princeton.edu/~sigm2002/papers/p152-cameron.pdf

An Adaptive Approach for Texture Segmentation by Multi-channel.. - Laine, Fan (1993) (Correct) (1 citation)
(DWPT) 29]30]which corresponds to a **general tree**-structured filter bank. The computational

levels. Figure 3 shows a general DWPF as a **binary tree** for a three level decomposition. In each case
ftp.cis.ufl.edu/pub/tech-reports/tr93/tr93-025.ps.Z

Loss-based Inference of Multicast Network Topology - Caceres, Duffield.. (1999) (Correct) (1 citation)
extension of the binary approach to treat more **general trees** was also proposed. The contributions of the the-the special case of the same estimator for **binary trees** in order to estimate loss on the common
gaia.cs.umass.edu/pub/Caceres99_cdc99.ps.Z

Efficient Wavelength Routing in Trees with Low-Degree .. - Auletta.. (1998) (Correct) (1 citation)
in Section 6. Finally we study the case of **general trees** and prove that the 5l=3 barrier can be broken (i.e.full bandwidth utilization) in case of **binary trees** using converters of constant degree. 1.
www.ceid.upatras.gr/~caragian/dimacs98.ps

Systematic Derivation of Tree Contraction Algorithms - Kiminori Matsuzaki Zhenjiang (Correct)
paper, we consider the parallelization of a **general tree** recursive function, called (tree) reduction, algorithm, the shunt contraction, can run on **binary trees** in logarithmic time to their size. However, it
www.ipl.t.u-tokyo.ac.jp/~hu/pub/cmpp04.pdf

A Fixed-Parameter Algorithm - For Minimum Quartet (Correct)
not the same set of sequenced proteins. In **general, tree** construction methods cannot take advantage of for this relationship is an evolutionary tree, a **binary tree** T in which the leaves are bijectively labeled
www-fs.informatik.uni-tuebingen.de/~niedermr/publications/./jcss04gn.ps.gz

IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS---II: ANALOG .. - Inductive.. (Correct)
A generic algorithm to insert repeaters in a **general tree** is presented in this section. The algorithm since any tree can be transformed into a **binary tree** by inserting zero impedance wires [8]13]
www.ee.rochester.edu:8080/~friedman/papers/TCASII.01.pdf

Tree-Structured GARCH Models - Francesco Audrino And (Correct)
performance. Our approach resembles more the **general tree** tting with the deviance criterion used by series. The approach relies on the idea of a **binary tree** where every terminal node parameterizes a
ftp.stat.math.ethz.ch/Research-Reports/Other-Manuscripts/buhlmann/treegarch.rev2.ps.gz

Nearest Common Ancestors : - Survey And New (Correct)
ancestor queries work using some mapping from a **general tree** to a complete **binary tree**. However, it is not for the problem is that a solution for complete **binary trees** is straightforward. Furthermore, for complete
dept-info.labri.fr/~gavoille/article/AGKR02.ps.gz

First 20 documents Next 20

Terms used **Preorder traversal**

Found **10** of **150,885**

Sort results by: relevance
Display results: expanded form

Save results to a Binder
Search Tips
Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 10 of 10

Relevance scale

1 Increasing robustness in global adaptive quadrature through interval selection heuristics
Henry D. Shapiro
May 1984 **ACM Transactions on Mathematical Software (TOMS)**, Volume 10 Issue 2
Full text available: pdf(1.14 MB) Additional Information: full citation, references, index terms, review

2 Low-power technology mapping for mixed-swing logic
Rob A. Rutenbar, L. Richard Carley, Roberto Zafalon, Nicola Dragone
August 2001 **Proceedings of the 2001 international symposium on Low power electronics and design**
Full text available: pdf(173.06 KB) Additional Information: full citation, references, index terms

3 Data abstraction, controlled iteration, and communicating processes
Alfs T. Berztiss
January 1980 **Proceedings of the ACM 1980 annual conference**
Full text available: pdf(641.21 KB) Additional Information: full citation, abstract, references, index terms

Iterators provide access to elements of an abstract structured object in some sequence. It is argued that parallel composition of iterators should be achieved implicitly by means of a generalized for loop rather than by use of mutually interacting coroutines. The generalized for loop employs controlled iteration, which is shown to be a powerful yet inexpensive construct. The generalized for loop is consistent with block structure, and, for program proof purposes, is much more tractable than ...

4 Problems from the 12th annual ACM programming contest
Lionel E. Deimel
December 1988 **ACM SIGCSE Bulletin**, Volume 20 Issue 4
Full text available: pdf(729.22 KB) Additional Information: full citation, citings, index terms

5 Active zones in CSG for accelerating boundary evaluation, redundancy elimination, interference detection, and shading algorithms
Jaroslaw R. Rossignac, Herbert B. Voelcker
November 1988 **ACM Transactions on Graphics (TOG)**, Volume 8 Issue 1
Full text available: pdf(2.67 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Solids defined by Boolean combinations of solid primitives may be represented in constructive solid geometry (CSG) as binary trees. Most CSG-based algorithms (e.g., for boundary evaluation, graphic shading, interference detection) do various forms of set-membership classification by traversing the tree associated with the solid. These algorithms usually generate intermediate results that do not contribute to the final result, and hence may be regarded as redundant and a source of inefficien ...

6 Compiling strictness into streams
C. V. Hall, D. S. Wise
October 1987 **Proceedings of the 14th ACM SIGACT-SIGPLAN symposium on Principles of programming languages**
Full text available: pdf(1.01 MB) Additional Information: full citation, references, citings, index terms

7 On the Generation of Binary Trees
Andrzej Proskurowski
January 1980 **Journal of the ACM (JACM)**, Volume 27 Issue 1
Full text available: pdf(117.26 KB) Additional Information: full citation, references, citings, index terms

8 Maps and descendant husband trees (abstract only)
Don Morrison
March 1985 **Proceedings of the 1985 ACM thirteenth annual conference on Computer Science**
Additional Information: full citation, abstract, index terms

MAP is an acronym for matching and permutation. A map is a finite set in which each member, x, has a spouse and a successor. The spouse function is a matching and the successor function is a permutation. The only tabulation required to describe a map is the successor function. The only primitive structure change is a successor swap. Maps are efficient hosts to many data structures, including linked lists, trees, forests and digraphs. In each guest, each mem ...

9 The under-appreciated unfold
Jeremy Gibbons, Geraint Jones
September 1998 **ACM SIGPLAN Notices , Proceedings of the third ACM SIGPLAN international conference on Functional programming**, Volume 34 Issue 1
Full text available: pdf(781.41 KB) Additional Information: full citation, abstract, references, citings, index terms

Folds are appreciated by functional programmers. Their dual, *unfolds*, are not new, but they are not nearly as well appreciated. We believe they deserve better. To illustrate, we present (indeed, we calculate) a number of algorithms for computing the *breadth-first traversal* of a tree. We specify breadth-first traversal in terms of *level-order traversal*, which we characterize first as a fold. The presentation as a fold is simple, but it is inefficient, and removing the ...

Keywords: anamorphism, breadth-first, co-induction, fold, functional programing, level-order, program calculation, traversal, unfold

10 On slicing programs with jump statements
Hiralal Agrawal
June 1994 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1994 conference on Programming language design and implementation**, Volume 29 Issue 6
Full text available: pdf(1.06 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Program slices have potential uses in many software engineering applications. Traditional slicing algorithms, however, do not work correctly on programs that contain explicit jump

statements. Two similar algorithms were proposed recently to alleviate this problem. Both require the flowgraph and the program dependence graph of the program to be modified. In this paper, we propose an alternative algorithm that leaves these graphs intact and uses a separate graph to store the additional requir ...

Results 1 - 10 of 10

Terms used **continuing traversal**

Found **34** of **150,885**

Sort results by: relevance

Display results: expanded form

Save results to a Binder

Search Tips

Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 1 - 20 of 34

Result page: **1** 2 next

Relevance scale

1 Computing complete answers to queries in the presence of limited access patterns

Chen Li

October 2003 **The VLDB Journal — The International Journal on Very Large Data Bases**, Volume 12 Issue 3

Full text available: pdf(347.40 KB) Additional Information: full citation, abstract, index terms

Abstract.In data applications such as information integration, there can be limited access patterns to relations, i.e., binding patterns require values to be specified for certain attributes in order to retrieve data from a relation. As a consequence, we cannot retrieve all tuples from these relations. In this article we study the problem of computing the *complete* answer to a query, i.e., the answer that could be computed if all the tuples could be retrieved. A query is *stable* if f ...

Keywords: Complete answers to queries, Limited access patterns to relations, Query stability

2 Recovering branches on the tree of life: an approximation algorithm

Paul Kearney, Ming Li, John Tsang, Tao Jiang

January 1999 **Proceedings of the tenth annual ACM-SIAM symposium on Discrete algorithms**

Full text available: pdf(957.72 KB) Additional Information: full citation, references, index terms

3 Advances in software and hardware synthesis techniques for DSP applications: Efficient mapping of hierarchical trees on coarse-grain reconfigurable architectures

F. Rivera, M. Sanchez-Elez, M. Fernandez, R. Hermida, N. Bagherzadeh

September 2004 **Proceedings of the 2nd IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis**

Full text available: pdf(316.12 KB) Additional Information: full citation, abstract, references, index terms

Reconfigurable architectures have become increasingly important in recent years. In this paper we present an approach to the problem of executing 3D graphics interactive applications onto these architectures. The hierarchical trees are usually implemented to reduce the data processed, thereby diminishing the execution time. We have developed a mapping scheme that parallelizes the tree execution onto a SIMD reconfigurable architecture. This mapping scheme considerably reduces the time penalty cau ...

Keywords: SIMD, computer graphics, hierarchical trees, multimedia, reconfigurable architectures

4

Low power SOCs and NOCs: High-level power analysis for on-chip networks
Noel Eisley, Li-Shiuan Peh
September 2004 **Proceedings of the 2004 international conference on Compilers, architecture, and synthesis for embedded systems**
Full text available: pdf(353.56 KB) Additional Information: full citation, abstract, references, index terms

As on-chip networks become prevalent in multiprocessor systems-on-a-chip and multi-core processors, they will be an integral part of the design flow of such systems. With power increasingly the primary constraint in chips, the tool chain in systems design, from simulation infrastructures to compilers and synthesis frameworks, needs to take network power into account, motivating the need for early-stage communication power analysis.While there has been substantial research in network performance ...

Keywords: link utilization, power analysis, simulation, systems-on-a-chip (SoC)

5 Depth-order point classification techniques for CSG display algorithms
Frederik W. Jansen
January 1991 **ACM Transactions on Graphics (TOG)**, Volume 10 Issue 1
Full text available: pdf(4.54 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Constructive Solid Geometry (CSG) defines objects as Boolean combinations (CSG trees) of primitive solids. To display such objects, one must classify points on the surfaces of the primitive solids with respect to the resulting composite object, to test whether these points lie on the boundary of the composite object or not. Although the point classification is trivial compared to the surface classification (i.e., the computation of the composite object), for CSG models with a large number o ...

6 ICICLE: groupware for code inspection
L. Brothers, V. Sembugamoorthy, M. Muller
September 1990 **Proceedings of the 1990 ACM conference on Computer-supported cooperative work**
Full text available: pdf(1.12 MB) Additional Information: full citation, abstract, references, citings, index terms

ICICLE1 ("Intelligent Code Inspection Environment in a C Language Environment") is a multifarious software system intended to augment the process of formal code inspection. It offers assistance in a number of activities, including knowledge-based analysis and annotations of source code, and computer supported cooperative discussion and finalization of inspectors' comments during inspection meetings. This paper reports the implementation of ICICLE and GroupWa ...

7 The TSE 310 expert prototype for the Airbus A310 commercial aircraft
Mehmet H. Goker, Selahattin Kuru
June 1990 **Proceedings of the third international conference on Industrial and engineering applications of artificial intelligence and expert systems - Volume 2**
Full text available: pdf(563.84 KB) Additional Information: full citation, abstract, references, index terms

Using available Troubleshooting Manuals, Mean Time Between Failure data, and the Maintenance Record of each aircraft the troubleshooting process of an Airbus A-310 is being automated by implementing an expert system (TSE_310). Starting off with the basic troubleshooting tree in the Troubleshooting Manual the program uses Last Removal Date and Mean Time Between Failure data to calculate failure probabilities of the parts connected to the relevant node of the troubleshooting tree, and by usin ...

8 Term rewriting with traversal functions
Mark G. J. Van Den Brand, Paul Klint, Jurgen J. Vinju
April 2003 **ACM Transactions on Software Engineering and Methodology (TOSEM)**, Volume 12 Issue 2
Full text available: pdf(541.97 KB) Additional Information: full citation, abstract, references, index terms

Term rewriting is an appealing technique for performing program analysis and program transformation. Tree (term) traversal is frequently used but is not supported by standard term rewriting. We extend many-sorted, first-order term rewriting with *traversal functions* that automate tree traversal in a simple and type-safe way. Traversal functions can be bottom-up or top-down traversals and can either traverse all nodes in a tree or can stop the traversal at a certain depth as soon as a match ...

Keywords: Automated tree traversal, term rewriting, types

9 Spherical parametrization and remeshing
Emil Praun, Hugues Hoppe
July 2003 **ACM Transactions on Graphics (TOG)**, Volume 22 Issue 3
Full text available: pdf(28.33 MB) Additional Information: full citation, abstract, references

The traditional approach for parametrizing a surface involves cutting it into charts and mapping these piecewise onto a planar domain. We introduce a robust technique for directly parametrizing a genus-zero surface onto a spherical domain. A key ingredient for making such a parametrization practical is the minimization of a stretch-based measure, to reduce scale-distortion and thereby prevent undersampling. Our second contribution is a scheme for sampling the spherical domain using uniformly sub ...

Keywords: geometry images, meshes, remeshing, texture mapping

10 Space Walking
Andrew J. Hanson, Hui Ma
October 1995 **Proceedings of the 6th conference on Visualization '95**
Full text available: pdf(1.10 MB) Publisher Site Additional Information: full citation, abstract

We propose an interactive method for exploring topological spaces based on the natural local geometry of the space. Examples of spaces appropriate for this visualization approach occur in abundance in mathematical visualization, surface and volume visualization problems, and scientific applications such as general relativity. Our approach is based on using a controller to choose a direction in which to "walk" a manifold along a local geodesic path. The method automatically generates orientation ...

Keywords: visualization, manifolds, mathematical visualization, interactive visualization

11 Utilizing first-order logic in query processing
Djamshid Asgari, Lawrence Henschen
January 1984 **Proceedings of the ACM 12th annual computer science conference on SIGCSE symposium**
Full text available: pdf(612.06 KB) Additional Information: full citation, abstract, references, index terms

The purpose of this research is to improve on the current membership algorithms and their applications in relational database model and information retrieval. A new technique proposed which makes up for deficiencies existing methods. This membership algorithm based on full first-order logic, extending Sagiv's prepositional calculus approach. Our technique does not assign one relation name (association) to all database dependencies like the representation of Nicolas. Finally, it may be used ...

12 Augmented Transition Networks as a design tool for personalized database systems
Alan L. Tharp
May 1978 **Proceedings of the 1st annual international ACM SIGIR conference on Information storage and retrieval**
Full text available: pdf(557.09 KB) Additional Information: full citation, abstract, references, citings, index terms

This paper illustrates the use of Augmented Transition Networks (ATNs) as a design tool for constructing document retrieval systems for those personalized applications which are too small or specialized to attract a commercial vendor. ATNs, which are explained in the context of this application, are used not only to improve the human/computer interface with the retrieval system but also to conceptually organize its structure.

13 Tamper-resistant whole program partitioning
Tao Zhang, Santosh Pande, Antonio Valverde
June 2003 **ACM SIGPLAN Notices , Proceedings of the 2003 ACM SIGPLAN conference on Language, compiler, and tool for embedded systems**, Volume 38 Issue 7
Full text available: pdf(444.16 KB) Additional Information: full citation, abstract, references, index terms

Due to limited available memory (of the order of Kilobytes) on embedded devices (such as smart cards), we undertake an approach of partitioning the whole program when it does not fit in the memory. The program partitions are downloaded from the server on demand into the embedded device just before execution. We devise a method of partitioning the code and data of the program such that no information regarding the control flow behavior of the program is leaked out. This property is called tamper ...

Keywords: mobile code, program partitioning, ramper resistance, smart card

14 Architectural power estimation and optimization: State-based power analysis for systems-on-chip
Reinaldo A. Bergamaschi, Yunjian W. Jiang
June 2003 **Proceedings of the 40th conference on Design automation**
Full text available: pdf(166.70 KB) Additional Information: full citation, abstract, references, index terms

Early power analysis for systems-on-chip (SoC) is crucial for determining the appropriate packaging and cost. This early analysis commonly relies on evaluating power formulas for all cores for multiple configurations of voltage, frequency, technology and application parameters, which is a tedious and error-prone process. This work presents a methodology and algorithms for automating the power analysis of SoCs. Given the power state machines for individual cores, this work defines the product pow ...

Keywords: power analysis, state exploration, systems-on-chip

15 A fresh look at combinator graph reduction
P. J. Koopman, P. Lee
June 1989 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1989 Conference on Programming language design and implementation**, Volume 24 Issue 7
Full text available: pdf(1.08 MB) Additional Information: full citation, abstract, references, citings, index terms

We present a new abstract machine for graph reduction called TIGRE. Benchmark results show that TIGRE's execution speed compares quite favorably with previous combinator-graph reduction techniques on similar hardware. Furthermore, the mapping of TIGRE onto conventional hardware is simple and efficient. Mainframe implementations of TIGRE provide performance levels exceeding those previously available on custom graph reduction hardware.

16 A generator of direct manipulation office systems
Scott E. Hudson, Roger King
July 1986 **ACM Transactions on Information Systems (TOIS)**, Volume 4 Issue 2
Full text available: pdf(2.45 MB) Additional Information: full citation, abstract, references, citings, index terms, review

A system for generating direct manipulation office systems is described. In these systems, the user directly manipulates graphical representations of office entities instead of dealing with these entities abstractly through a command language or menu system. These systems employ a new semantic data model to describe office entities. New techniques based on

attribute grammars and incremental attribute evaluation are used to implement this data model in an efficient manner. In addition, the s ...

17 Session 10D: digital and analog test generation: Deterministic test pattern generation techniques for sequential circuits
Ilker Hamzaoglu, Janak H. Patel
November 2000 **Proceedings of the 2000 IEEE/ACM international conference on Computer-aided design**
Full text available: pdf(57.38 KB) Additional Information: full citation, abstract, references

This paper presents new test generation techniques for improving the average-case performance of the iterative logic array based deterministic sequential circuit test generation algorithms. To be able to assess the effectiveness of the proposed techniques, we have developed a new ATPG system for sequential circuits, called ATOMS, and we have incorporated these techniques into the test generator. ATOMS achieved very high fault coverages in a short amount of time for the ISCAS89 sequential benchma ...

18 Parallel volume rendering: An interleaved parallel volume renderer with PC-clusters
Antonio Garcia, Han-Wei Shen
September 2002 **Proceedings of the Fourth Eurographics Workshop on Parallel Graphics and Visualization**
Full text available: pdf(1.46 MB) Additional Information: full citation, abstract, references

Parallel Volume Rendering has been realized using various load distribution methods that subdivide either the screen, called image-space partitioning, or the volume dataset, called object-space partitioning. The major advantages of image-space partitioing are load balancing and low communication overhead, but processors require access to the full volume in order to render the volume with arbitrary views without frequent data redistributions. Subdividing the volume, on the other hand, provides st ...

19 Session 3: High performance dynamic lock-free hash tables and list-based sets
Maged M. Michael
August 2002 **Proceedings of the fourteenth annual ACM symposium on Parallel algorithms and architectures**
Full text available: pdf(238.11 KB) Additional Information: full citation, abstract, references, citings, index terms

Lock-free (non-blocking) shared data structures promise more robust performance and reliability than conventional lock-based implementations. However, all prior lock-free algorithms for sets and hash tables suffer from serious drawbacks that prevent or limit their use in practice. These drawbacks include size inflexibility, dependence on atomic primitives not supported on any current processor architecture, and dependence on highly-inefficient or blocking memory management techniques.Building on ...

20 A computer science courseware factory
Michael J. Barnes, R. Hsu, N. Hsu, T. Sun, T. Nguyen, G. Haus, P. D. Smith
February 1986 **ACM SIGCSE Bulletin , Proceedings of the seventeenth SIGCSE technical symposium on Computer science education**, Volume 18 Issue 1
Full text available: pdf(850.68 KB) Additional Information: full citation, abstract, references, index terms

A model for computer science courseware development within universities is proposed. This model asserts that masters level graduate and undergraduate students are a valuable software development resource that can be utilized to design, prototype, field test, and refine quality courseware. To support this contention four prototype courseware packages are used to briefly describe the Visible Algorithms project at Northridge. These packages provide tutorials on, and graphic animations of: link ...

Useful downloads: Adobe Acrobat QuickTime Windows Media Player Real Player

Terms used **dynamic binary tree**

Found **6** of **150,885**

Sort results by: relevance

Display results: expanded form

Save results to a Binder

Search Tips

Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 1 - 6 of 6

Relevance scale

1 Representing dynamic binary trees succinctly

J. Ian Munro, Venkatesh Raman, Adam J. Storm

January 2001 **Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms**

Full text available: pdf(627.65 KB) Additional Information: full citation, abstract, references, citings, index terms

We introduce a new updatable representation of binary trees. The structure requires the information theoretic minimum 2*n* + *&Ogr;*(*n*) bits and supports basic navigational operations in constant time and subtree size in *&Ogr;*(lg *n*). In contrast to the linear update costs of previously proposed succinct representations, our representation supports updates in *&Ogr;*(lg^2 *n*) amortized time.

2 Dynamic parallel tree contraction (extended abstract)

John H. Reif, Stephen R. Tate

August 1994 **Proceedings of the sixth annual ACM symposium on Parallel algorithms and architectures**

Full text available: pdf(960.71 KB) Additional Information: full citation, abstract, references, index terms

Parallel tree contraction has been found to be a useful and quite powerful tool for the design of a wide class of efficient graph algorithms. We propose a corresponding technique for the parallel solution of incremental problems. As our computational model, we assume a variant of the CRCW PRAM where we can dynamically activate processors by a forking operation. We consider a dynamic binary tree T of ≤ n nodes and unbounded depth. We describe a ...

3 Cache oblivious search trees via binary trees of small height

Gerth Stølting Brodal, Rolf Fagerberg, Riko Jacob

January 2002 **Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete algorithms**

Full text available: pdf(1.10 MB) Additional Information: full citation, abstract, references, citings

We propose a version of cache oblivious search trees which is simpler than the previous proposal of Bender, Demaine and Farach-Colton and has the same complexity bounds. In particular, our data structure avoids the use of weight balanced *B*-trees, and can be implemented as just a single array of data elements, without the use of pointers. The structure also improves space utilization.For storing *n* elements, our proposal uses (1 + ε)*n* times the element size of memory, ...

4 Linear-time compression of bounded-genus graphs into information-theoretically optimal number of bits

Hsueh-I Lu

January 2002 **Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete**

algorithms

Full text available: pdf(217.75 KB) Additional Information: full citation, abstract, references, citings

This extended abstract summarizes a new result for the *graph compression* problem, addressing how to *compress* a graph *G* into a binary string *Z* with the requirement that *Z* can be *decoded* to recover *G*. Graph compression finds important applications in 3D model compression of Computer Graphics [12, 17-20] and compact routing table of Computer Networks [7]. For brevity, let a ¦Ð-*graph* stand for a graph with property ¦ ...

5 Back to direct style II: first-class continuations

Olivier Danvy, Julia L. Lawall

January 1992 **ACM SIGPLAN Lisp Pointers , Proceedings of the 1992 ACM conference on LISP and functional programming**, Volume V Issue 1

Full text available: pdf(1.02 MB) Additional Information: full citation, abstract, references, citings, index terms

We continue to investigate the direct-style transformation by extending it to programs requiring call-with-current-continuation (a.k.a. call/cc). The direct style (DS) and the continuation-passing style (CPS) transformations form a Galois connection. This pair of functions has a place in the programmer's toolbox—yet we are not aware of the existence of any other DS transformer. Starting from our DS transformer towards pure, call-by-value function ...

6 Performance of height-balanced trees

P. L. Karlton, S. H. Fuller, R. E. Scroggs, E. B. Kaehler

January 1976 **Communications of the ACM**, Volume 19 Issue 1

Full text available: pdf(490.66 KB) Additional Information: full citation, abstract, references, citings

This paper presents the results of simulations that investigate the performance of height-balanced (HB[k]) trees. It is shown that the only statistic of HB[1] trees (AVL trees) that is a function of the size of the tree is the time to search for an item in the tree. For sufficiently large trees, the execution times of all procedures for maintaining HB[1] trees are independent of the size of the tree. In particular, an average of .465 restructures are required per insertion, with an average ...

Keywords: AVL trees, HB[k] trees, balanced trees, information storage and retrieval, searching

Results 1 - 6 of 6